

株式会社テクノプロジェクト様

運用設計導入コース (第2回)

運用課題の分析

本編

運用設計ラーニング

2022-12-22

本編 運用課題の分析

運用課題の分析に必要なこと

講師が「運用あるある」で気付いたこと (第1回)

問題点1: 「課題」への対応を「点」に対して行なっていた

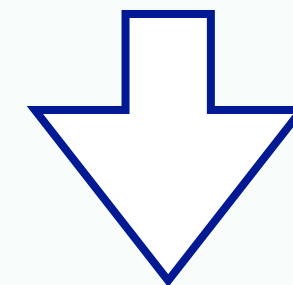
問題点2: 過去の延長線上で課題を解決しようとしていた

問題点3: 全ての「課題」が発生後の対応になっていた

分析ポイント1: 「課題」を「面」で捉えて分析

問題点1: 「課題」への対応を「点」に対して行なっていた

「課題」を「面」で捉えて分析する。



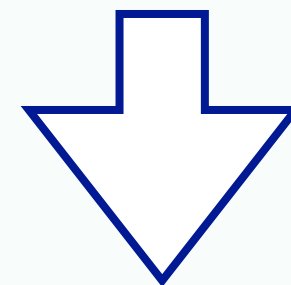
過去の解決事例を活用し、隣接業務や類似業務も含めて分析する。

課題の収集と蓄積

分析ポイント2: 「課題」を設計で解決することを前提に分析

問題点2: 過去の延長線上で課題を解決しようとしていた

「課題」は「設計で解決」(根本解決)することを前提に分析する。



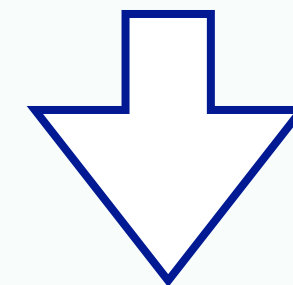
「課題」は原則として設計にフィードバックし、設計側で解決する。

課題を設計に共有

分析ポイント3: 「課題」を事後的・予防的双方の観点から分析

問題点3: 全ての「課題」が発生後の対応になっていた

「課題」を事後的・予防的双方の観点から分析する。



過去の解決課題の事例を活用して予測・予防する。

課題の継続的な分析

まとめ: 課題分析のポイント

課題の収集と蓄積

過去の解決事例を活用し、隣接業務や類似業務も含めて分析する。

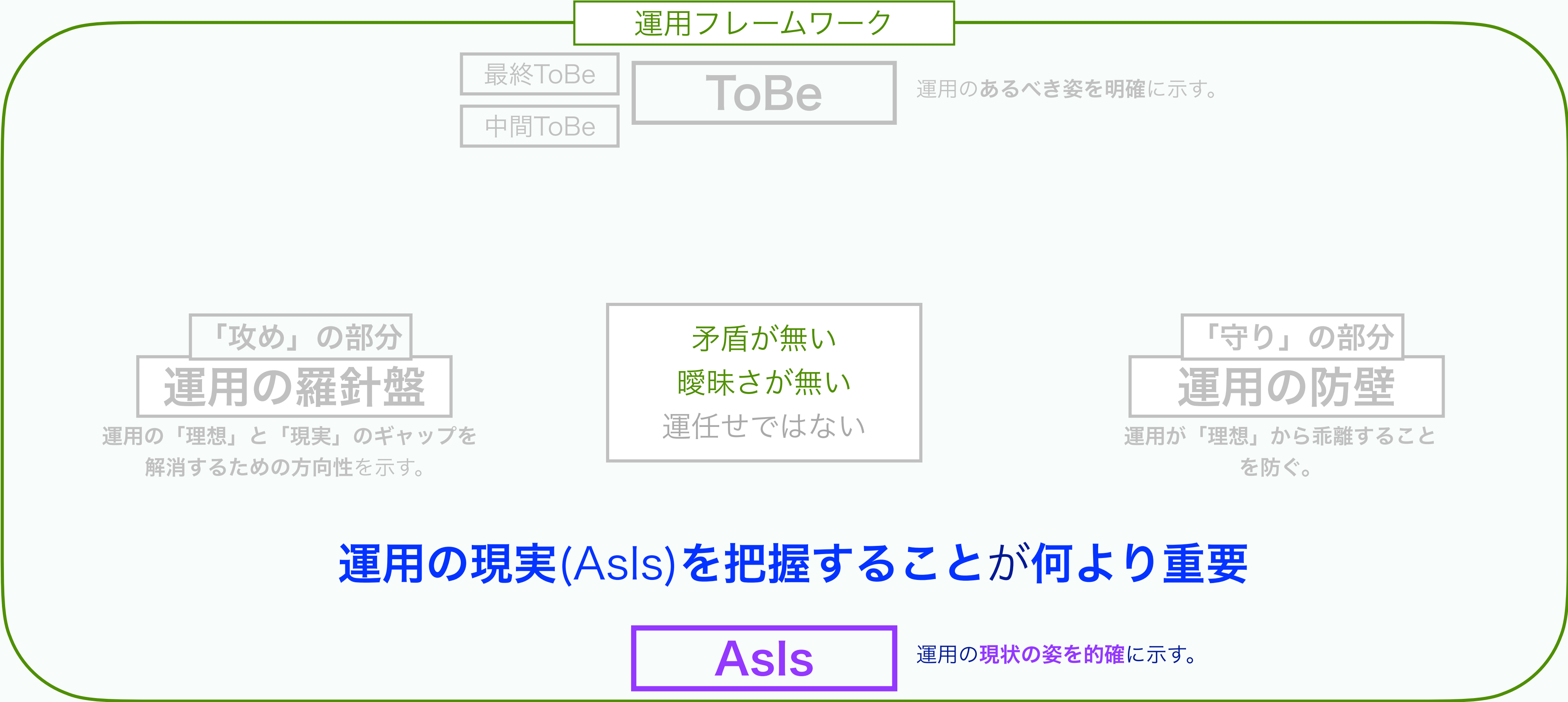
課題を設計に共有

「課題」は原則として設計にフィードバックし、設計側で解決する。

課題の継続的な分析

過去の解決課題の事例を活用して予測・予防する。

運用課題分析の前提: 運用現場の「現実」の把握



既に運用が動いている場合は、一番最初にやるべきこと。

運用課題の分析に必要なこと = 運用現場のAsIsの把握

運用の現実(AsIs)を把握することが何より重要

「運用課題」というネガティブなAsIsだけではなく、運用のポジティブなAsIsも含めて把握すべき

運用現場のAsIsの収集と蓄積

運用現場のAsIsを運用現場と設計側で共有

運用現場のAsIsの継続的な分析

本編 運用課題の分析

Aslsを知るためのサブフレームワーク

運用現場のAslsの把握

運用の現実(Asls)を把握することが何より重要

「運用課題」というネガティブなAslsだけでなく、運用のポジティブなAslsも把握すべき

運用現場のAslsの収集と蓄積

Aslsのインプット

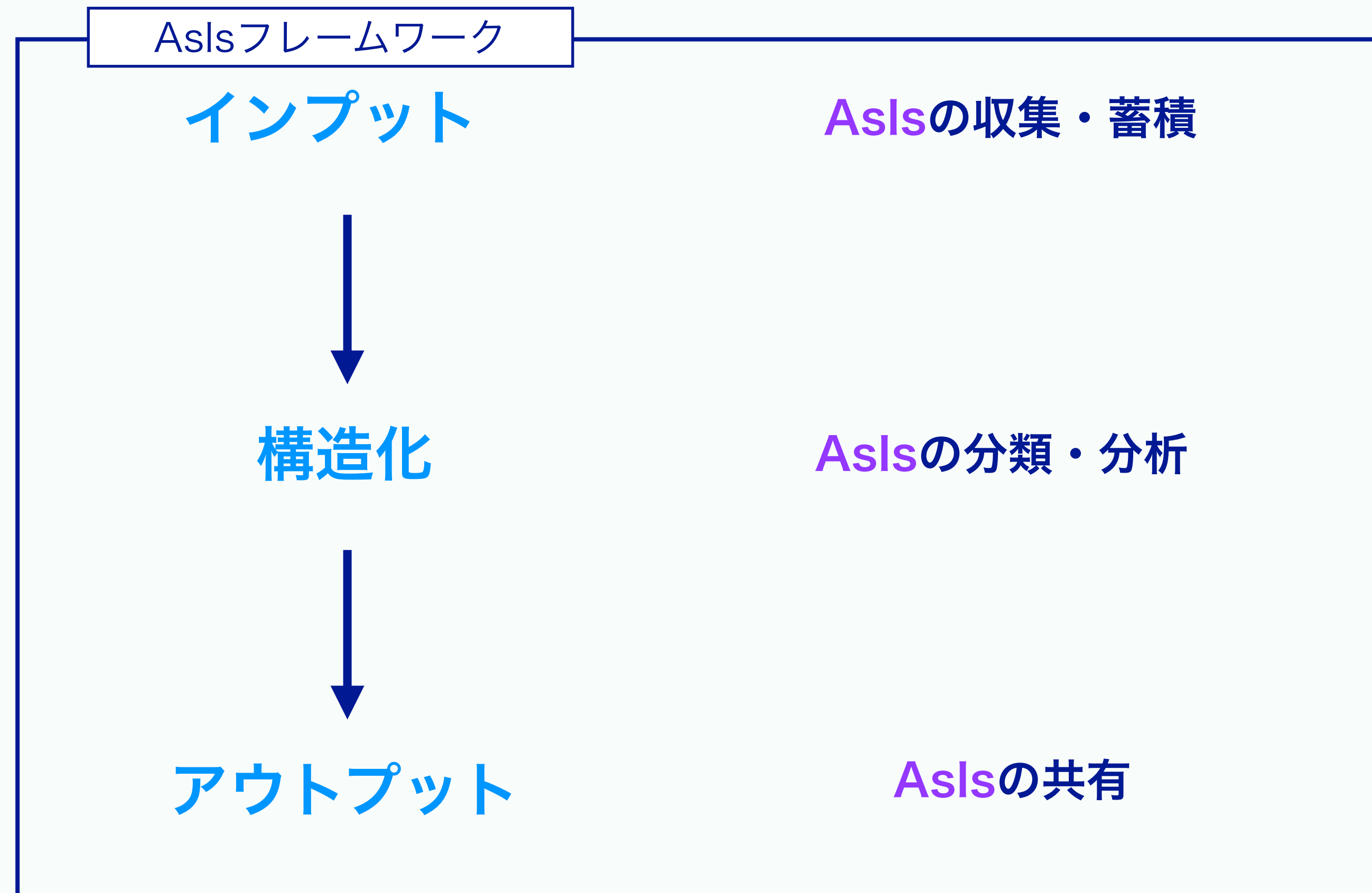
運用現場のAslsを運用現場と設計側で共有

Aslsのアウトプット

運用現場のAslsの継続的な分析

Aslsの構造化 (分析)

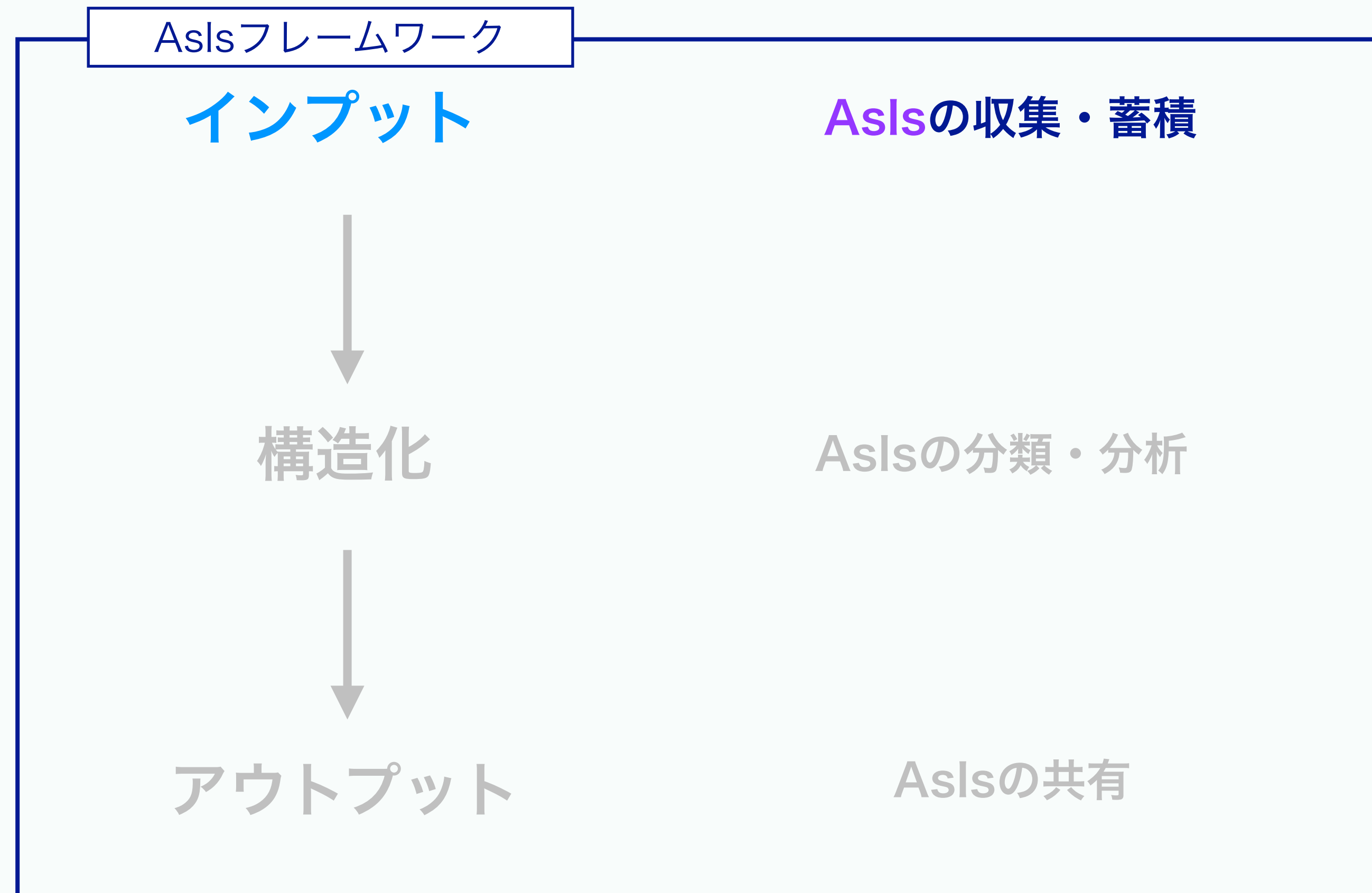
Aslsを知るためのサブフレームワーク



Aslsを知るためのサブフレームワーク

1. インプット

Aslsを知るためのサブフレームワーク

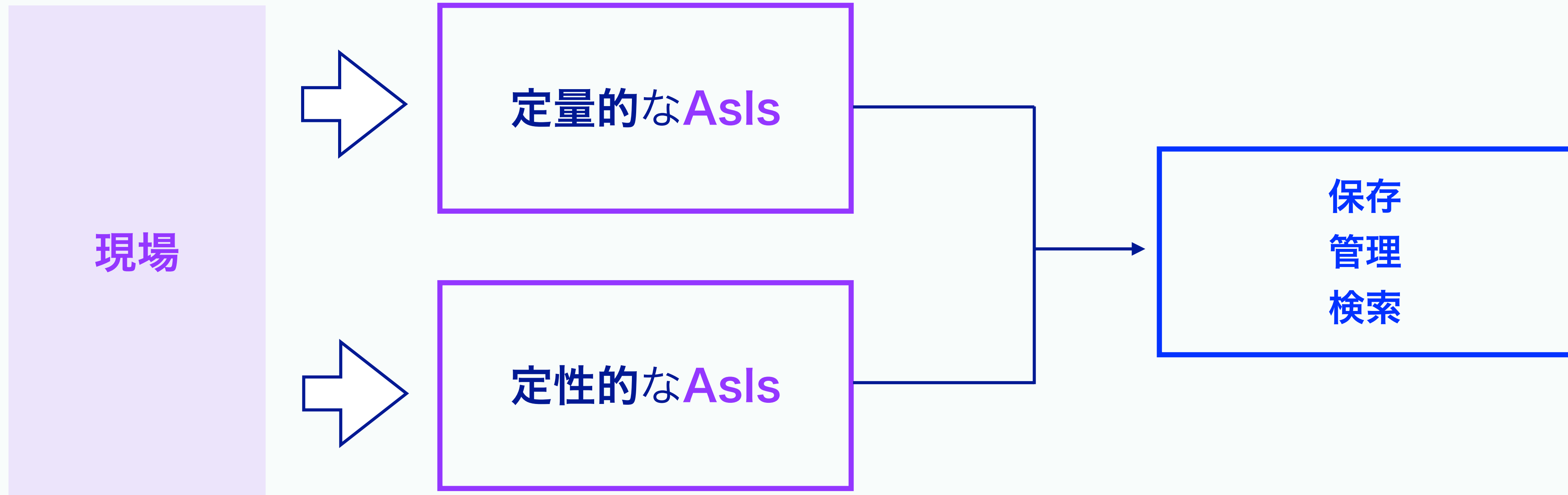


Aslsのインプットの全体像

インプット

Aslsの収集

Aslsの蓄積



Aslsのインプット (収集: 定量的Asls)

インプット



AsIsのインプット (収集: 定量的AsIs)

インプット AsIsの収集

現場の定量的な情報を収集する仕組みが必要。

Step1

収集対象の選定

収集する定量的情報を選定する。

Step2

一次情報の決定

収集する定量的情報に必要な一次情報を決定する。

Step3

データフローの整備

一次情報を定量的情報に変換するためのデータフローを整備する。

データフローの整備により、定量的情報の自動集計への下地を整える。

Aslsのインプット (収集: 定性的Asls)

インプット



AsIsのインプット (収集: 定性的AsIs)

インプット AsIsの収集

現場の定性的な情報が上がってくる、上がってきやすい仕組みが必要。

Step1

三現主義

マネージャやアーキテクトが定期的に現場に話を聞きに行く。

Step2

場の整備

気軽にヒヤリハットの存在を共有する場を整備する。

Step3

文化の醸成

最初にミスした人やネガティブ情報を上げた人を評価する。

「現場に聞きに行くこと」から初めて、仕組み化、文化の形成をしていく。

Aslsのインプット (蓄積)

インプット

Aslsの収集

Aslsの蓄積



AsIsのインプット (蓄積)

インプット AsIsの蓄積

収集したAsIsを蓄積する仕組みが必要。

保存

収集したAsIsを永続的に保存する。(内容によっては数年で破棄)

管理

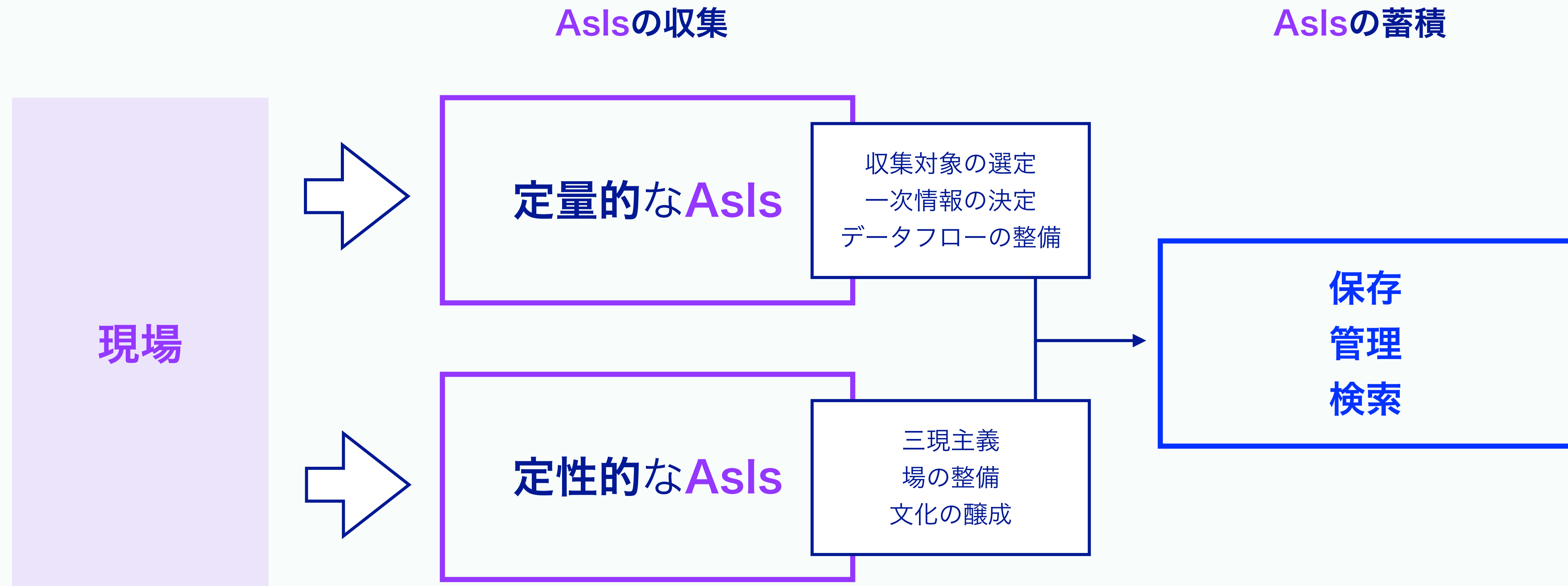
メタ情報(情報の管理情報)の管理をする。

検索

蓄積されたAsIsを効率的に検索・参照する。

まとめ: Aslsのインプット

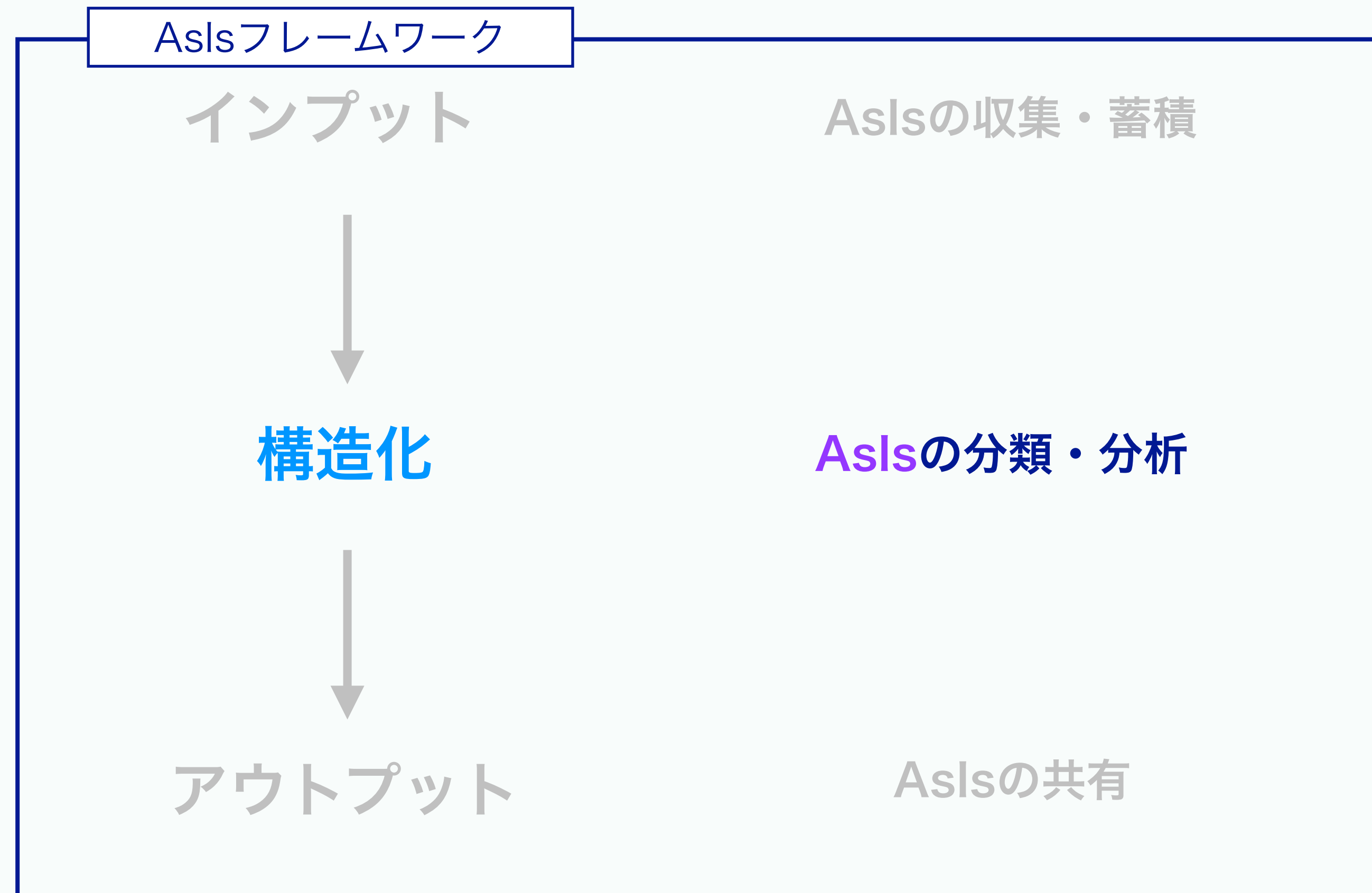
インプット



Aslsを知るためのサブフレームワーク

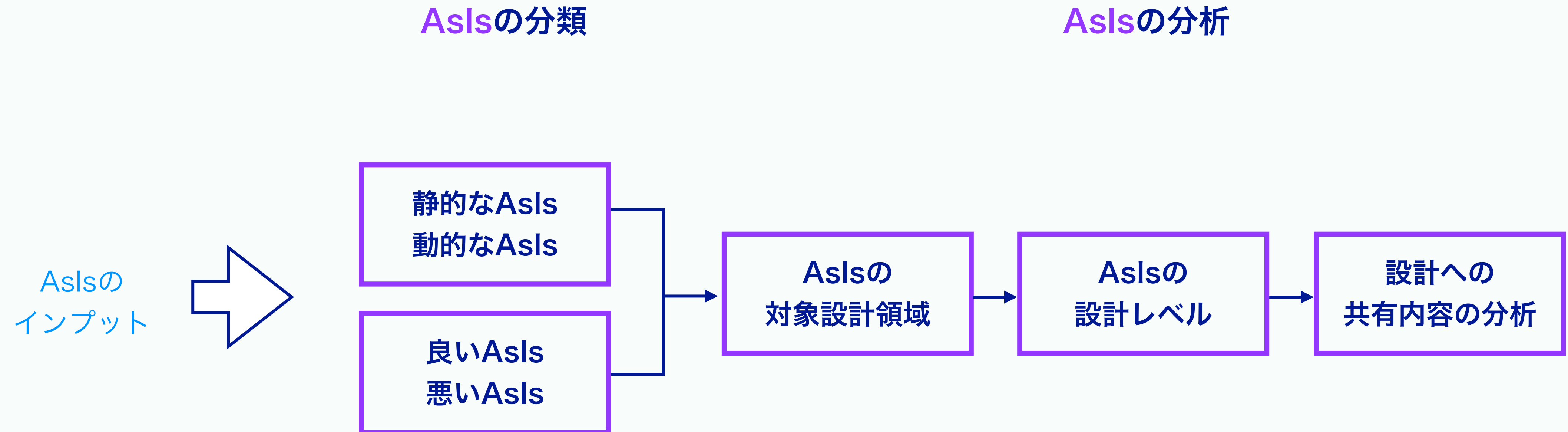
2. 構造化

Aslsを知るためのサブフレームワーク



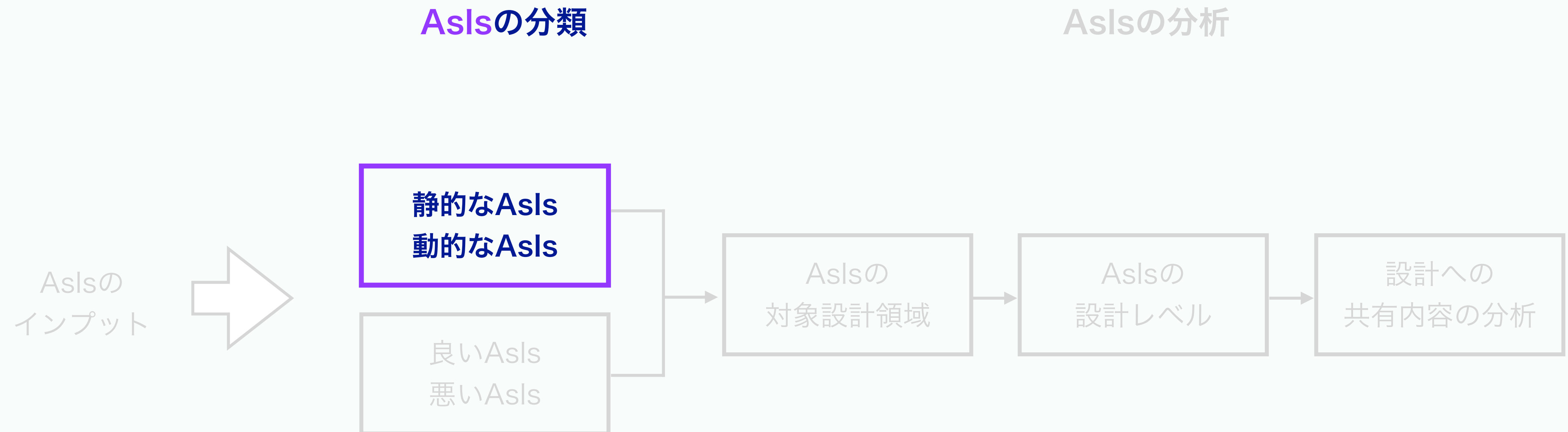
Aslsの構造化の全体像

構造化



Aslsの構造化 (分類: 静的なAslsと動的なAsls)

構造化



Aslsの構造化 (分類: 静的なAslsと動的なAsls)

構造化

Aslsの分類

Aslsに対する説明責任がどちらにあるかを分類する。

静的なAsls 現在の設計

「であるはず。」 **設計者の認識**

現在の設計の説明責任を担う。

設計に曖昧さ、矛盾、運任せがあった場合は修正する。

動的なAsls 現在、運用現場で起きていること、起きている変化

「である。」 「となりつつある。」 **現場職の認識**

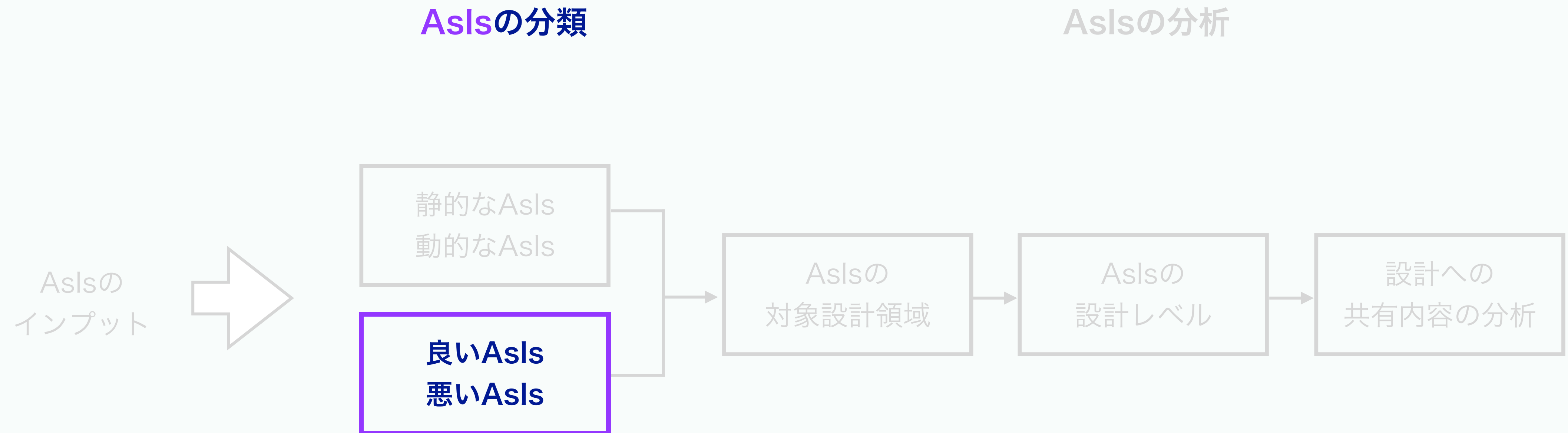
現場における現実や変化の説明責任を担う。

設計と現実の差異があった場合は情報を設計側に共有する。

自己の認識を言語化して、他者に伝えることができる「説明能力」が全員に求められる。

Aslsの構造化 (分類: 良いAslsと悪いAsls)

構造化



AsIsの構造化 (分類: 良いAsIsと悪いAsIs)

構造化

AsIsの分類

AsIsが運用現場にとって、良いものなのか、悪いものなのかを分類する。

良いAsIs ToBeに近い状態にある。

- ・ 静的に設計通り。かつ、動的に現実に合致。

仮にToBeに近くなくても「現段階の設計」には沿っている。

課題が発生しにくい。
課題が発生しても対処しやすい。

悪いAsIs ToBeから遠い状態にある。

- ・ 設計通りに実装されていない。(違反実装)
- ・ 設計が現実から乖離。(乖離設計)
- ・ そもそも設計がされていない。(未設計)

矛盾、曖昧、運任せが存在している。

これが自体が「課題」である。
更に、新たな課題の要因になっている。

多くの場合、悪いAsIsが課題の発生源。

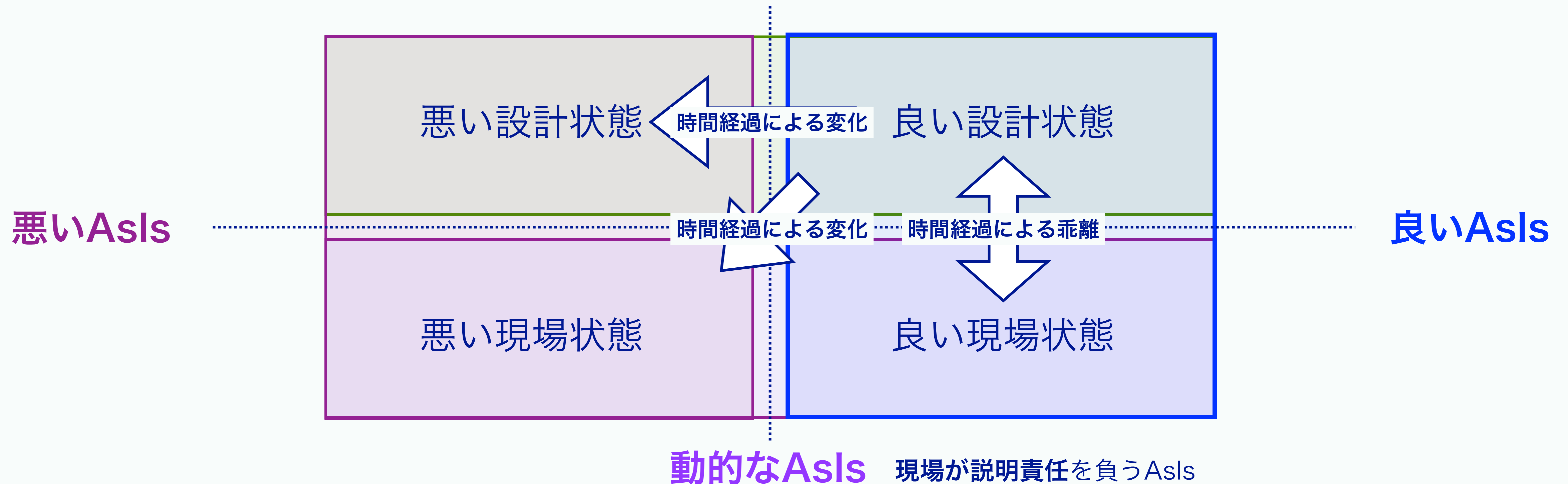
Aslsの構造化 (分類: まとめ)

構造化

Aslsの分類

Aslsの分類によって、Aslsが関連する設計や現場の状態の善し悪しを把握する。

静的なAsls 設計側が説明責任を負うAsls



Aslsの構造化 (分析: どの設計領域のAslsか)

構造化

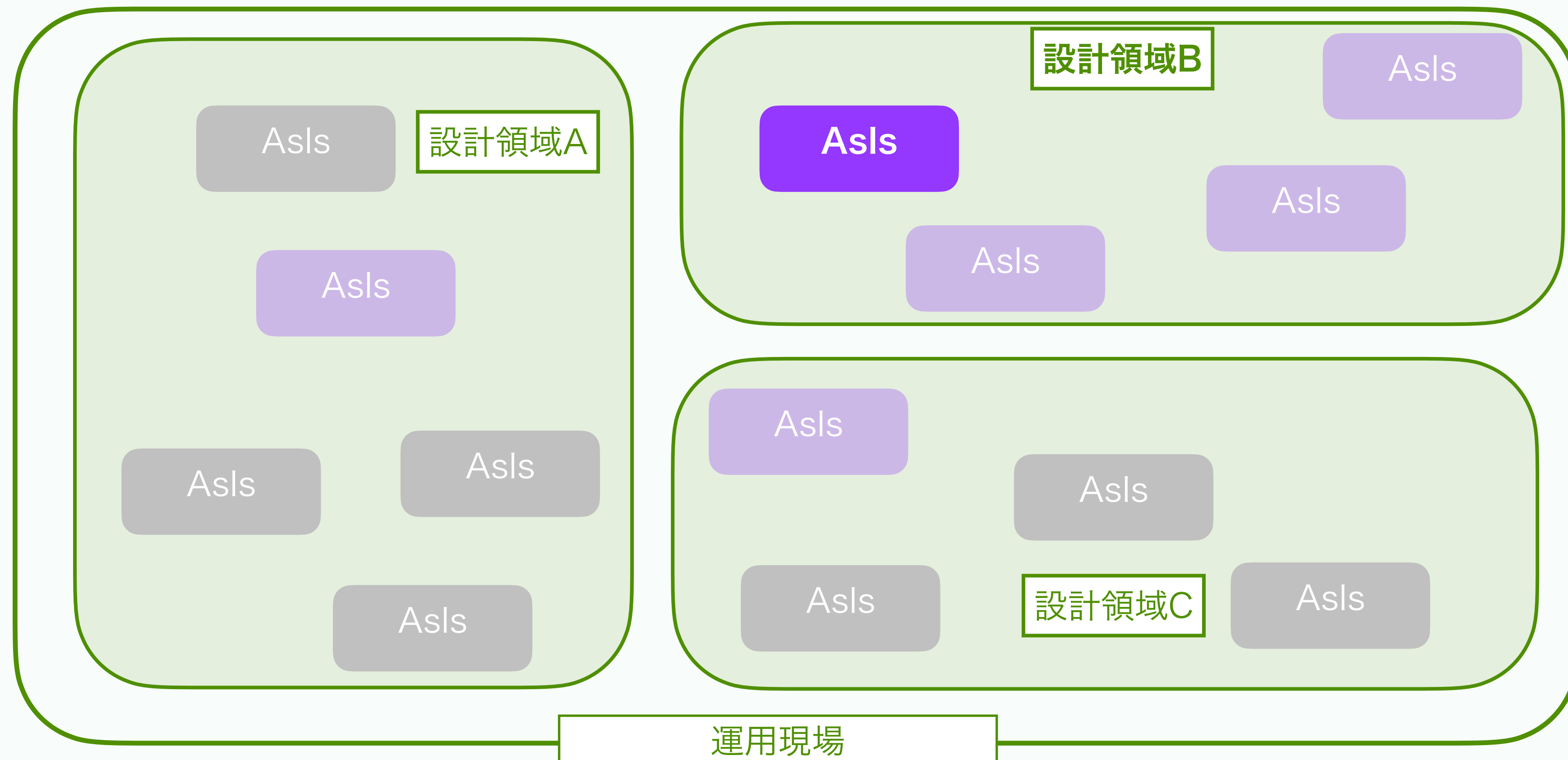


Aslsの構造化 (分析: どの設計領域のAslsか)

構造化

Aslsの分析

Aslsが関連する設計領域や影響する範囲を分析する



Aslsの構造化 (分析: Aslsレベル)

構造化



Aslsの構造化 (分析: Aslsレベル)

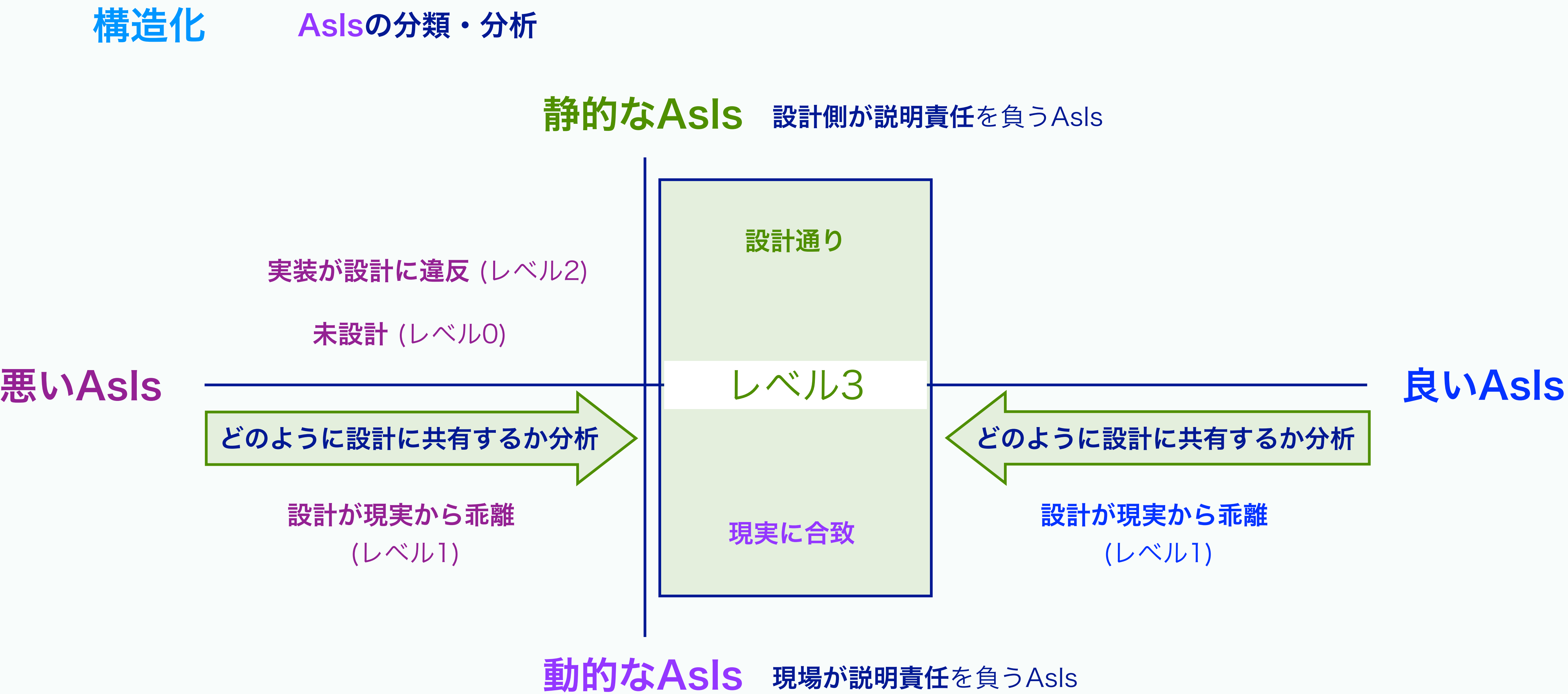
構造化 Aslsの分析

Aslsにより判明した設計の状態(レベル)を分析する

レベル0	未設計	設計がされていない状態や曖昧な状態が判明 環境や前提条件の変化により、いつでも発生し得る。
レベル1	設計が現実から乖離	設計が現実から乖離している状態が判明 現実の変化により、いつでも発生し得る。
レベル2	実装が設計に違反	設計が現実在即していても、実装が設計から乖離している状態が判明 設計の変化により、いつでも発生し得る。
レベル3	設計と現実が合致	設計が現実在即し、実装も設計に即している状態が判明

現実の変化により、Aslsレベルは常に下がっていく

参考: Aslsの分類とAslsレベル



Aslsの構造化 (分析: 共有内容の分析)

構造化



AsIsの構造化 (分析: 共有内容の分析)

構造化

AsIsの分析

「課題」を「設計で解決」(根本解決)するための分析

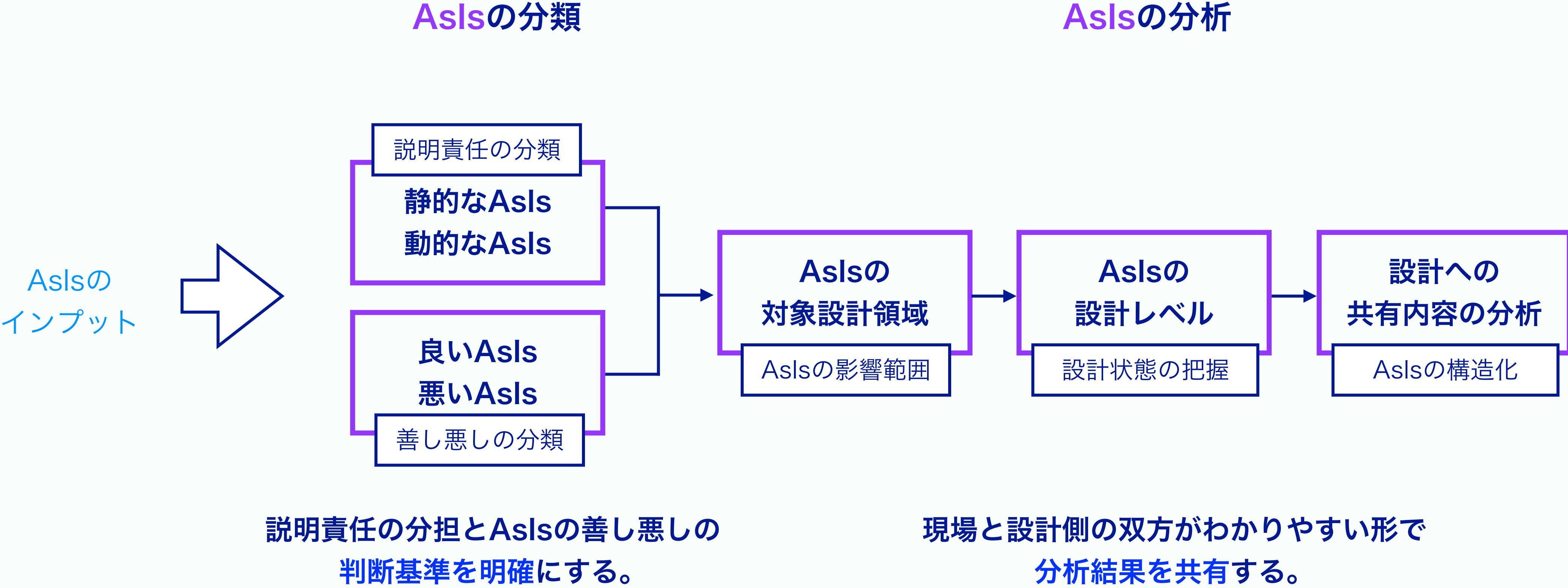
良いAsIsの場合は、「どのように設計に取り込むべきか」を分析する。

- ・ **AsIsは常に変わる**。インプットで上がってきた新鮮な情報をすぐに活かす。
- ・ **客観的に情報を見る**。定性的、定量的のバランス良く見る。
- ・ **数字で表現する**。やはり定量的に評価できるとより良い。

現場と設計側の双方が理解しやすい形で、分析結果を示すことが大切。

まとめ: Aslsの構造化

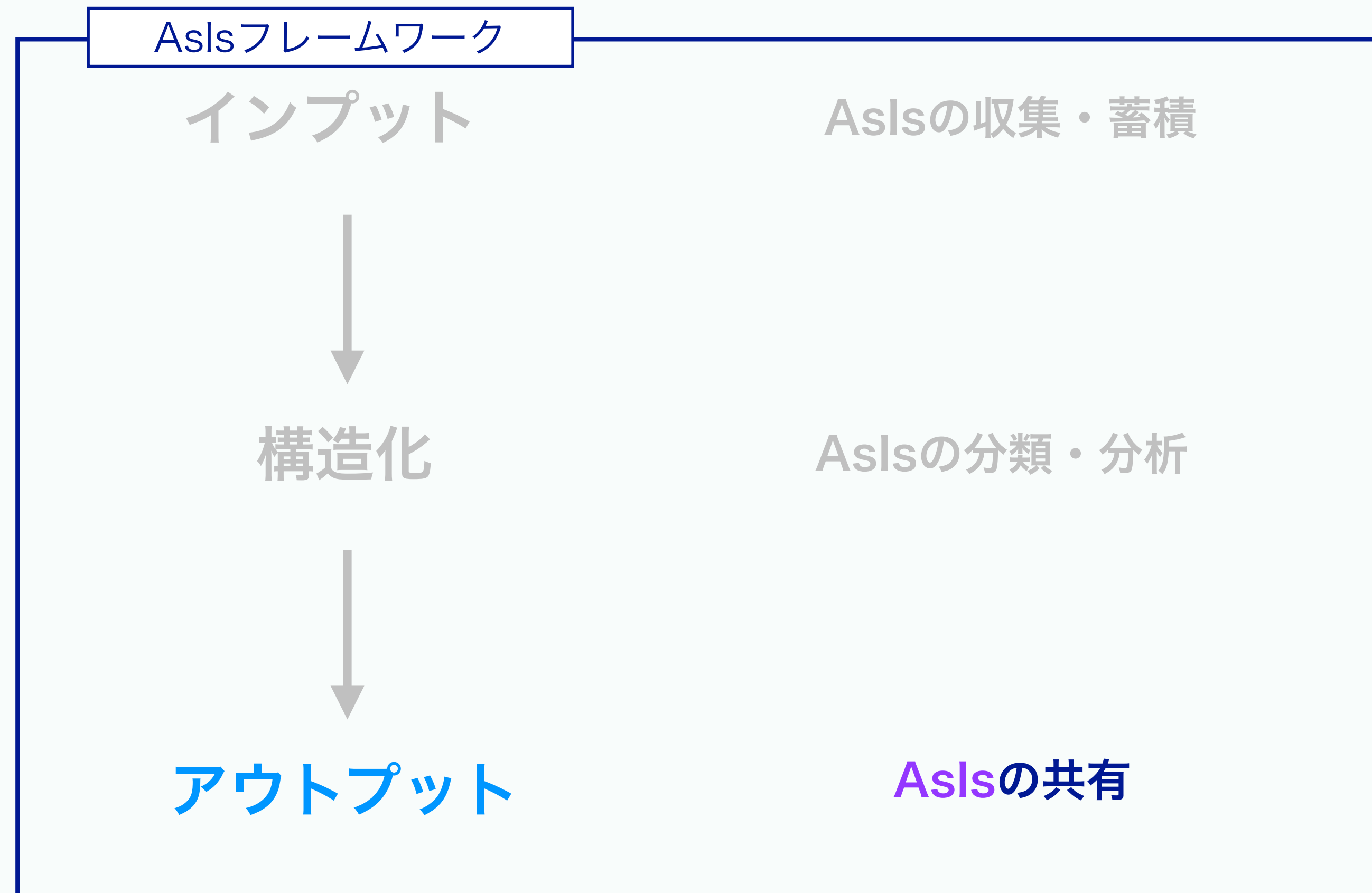
構造化



Aslsを知るためのサブフレームワーク

3. アウトプット

Aslsを知るためのサブフレームワーク



Aslsのアウトプットの全体像

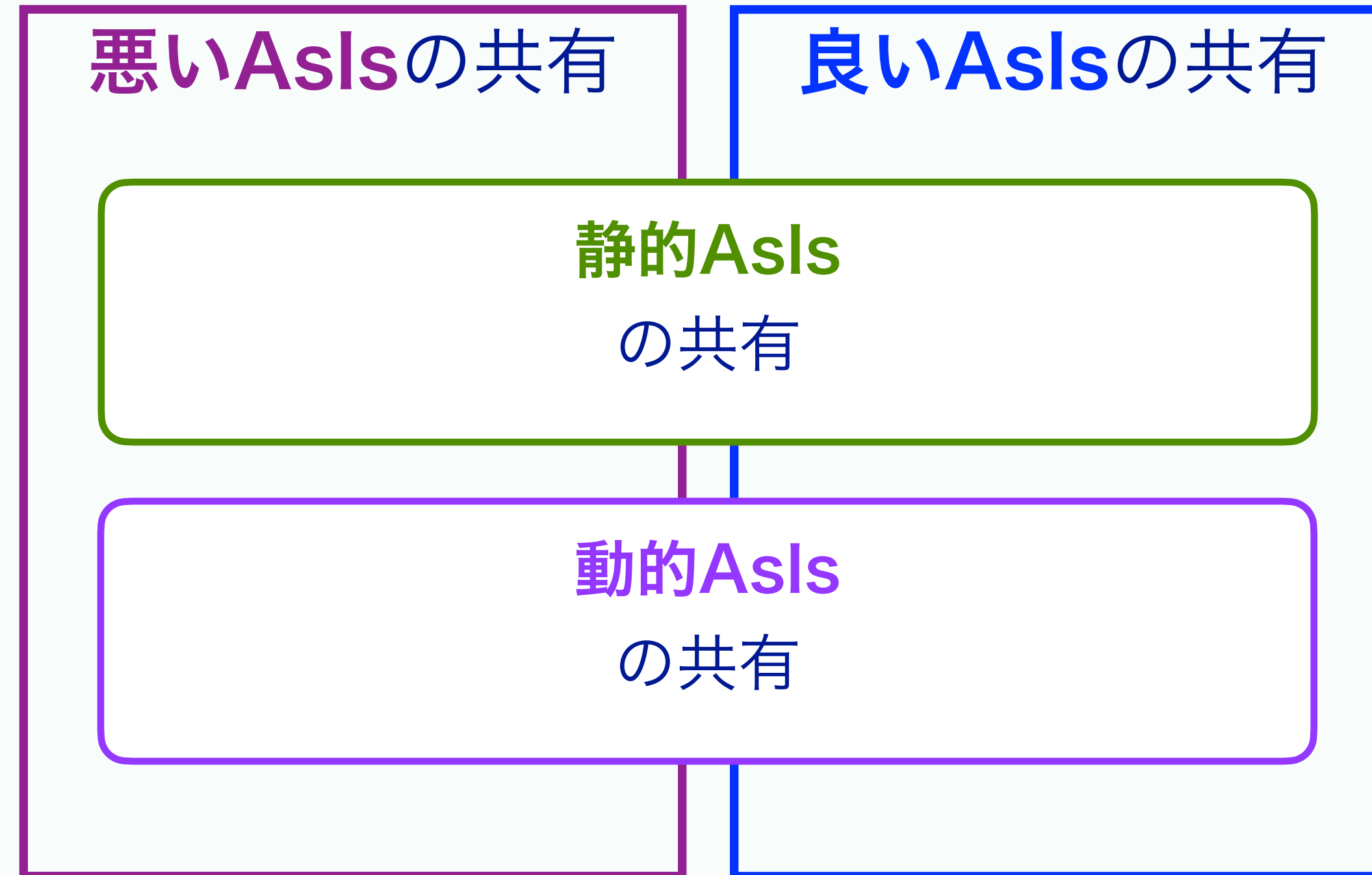
アウトプット



Aslsのアウトプット (良いAsls vs. 悪いAsls)

アウトプット

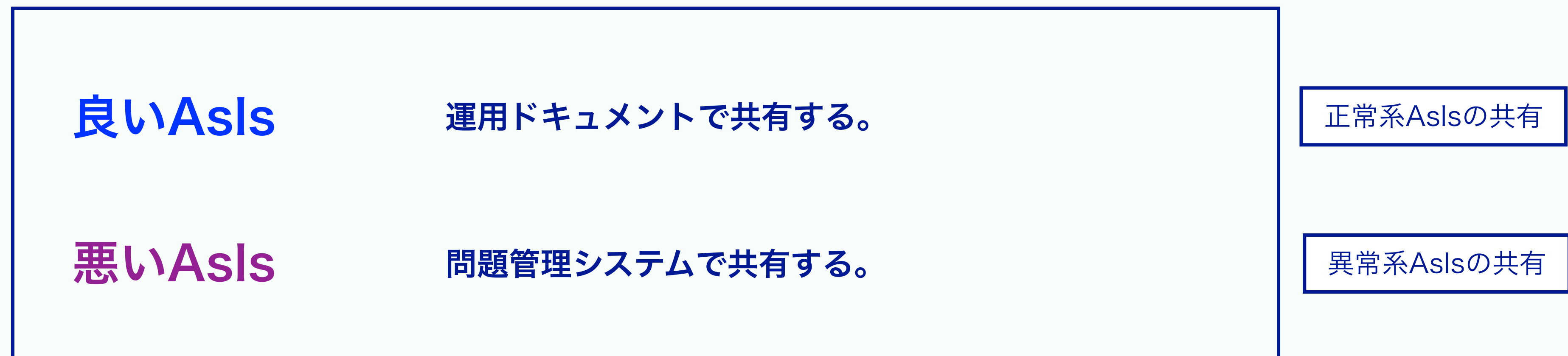
Aslsの
構造化



Aslsのアウトプット (良いAsls vs. 悪いAsls)

アウトプット Aslsの共有

運用に関与する人なら誰でも見えるところに、Aslsを公開する。



良いAslsを正常系、悪いAslsを異常系として、明確に分離して公開する。

Aslsのアウトプット (静的Asls vs. 動的Asls)

アウトプット



Aslsのアウトプット (静的Asls vs. 動的Asls)

アウトプット Aslsの共有

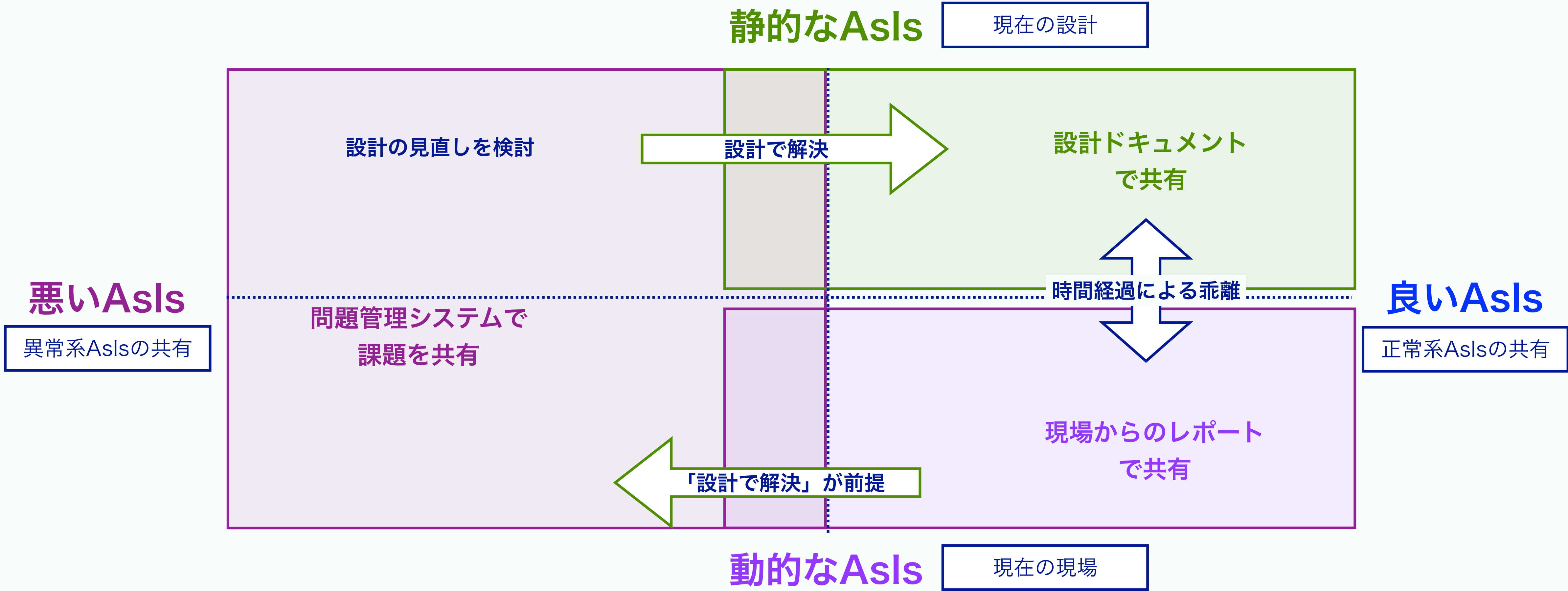
運用に関与する人なら誰でも見えるところに、Aslsを公開する。

静的なAsls	設計ドキュメントを作成・更新し、正確に共有する。	現在の設計や実装
動的なAsls	現場の声をレポートし、的確に共有する。	現在の現場

誰の視点でそのAslsを捉えているか、立場を明確にする。

参考: Aslsのアウトプットの仕組み化イメージ

アウトプット Aslsの共有



留意点: 悪いAsIsの扱い

アウトプット

AsIsの共有

設計変更せずに、いきなり対応しようとするしない。

良いAsIs

設計ドキュメント・実装ドキュメントで共有する。

正常系AsIsの共有

悪いAsIs

問題管理システムで共有する。

異常系AsIsの共有

いきなり対応しようとするしない。

現状の設計やToBeとの整合性を考え、必ず再設計する必要がある。

まとめ: Aslsのアウトプット

アウトプット



補講: 運用の課題分析の実装イメージ

補講: 運用の課題分析の実装イメージ

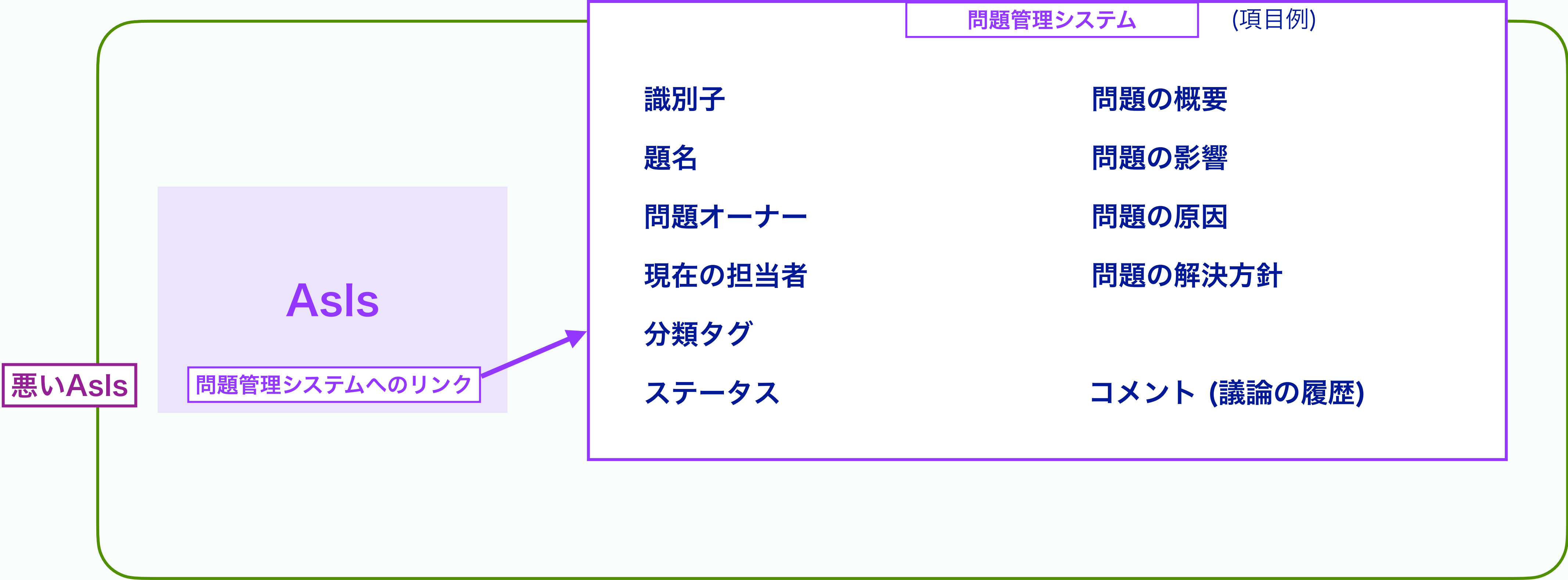
設計・実装ドキュメント、問題管理システムへの動線を
運用ダッシュボードで明示する



ドキュメントにも、問題管理システムにも、アクセスが容易であることが重要。

補講: 運用の課題分析の実装イメージ

「悪いAsls」は問題管理システムで問題管理をしていく。



問題管理システムにアクセスが容易であることが重要。

まとめ

講義のふりかえり

運用課題の分析に必要なこと

運用の現実(AsIs)を把握することが何より重要

「運用課題」というネガティブなAsIsだけではなく、運用のポジティブなAsIsも含めて把握すべき

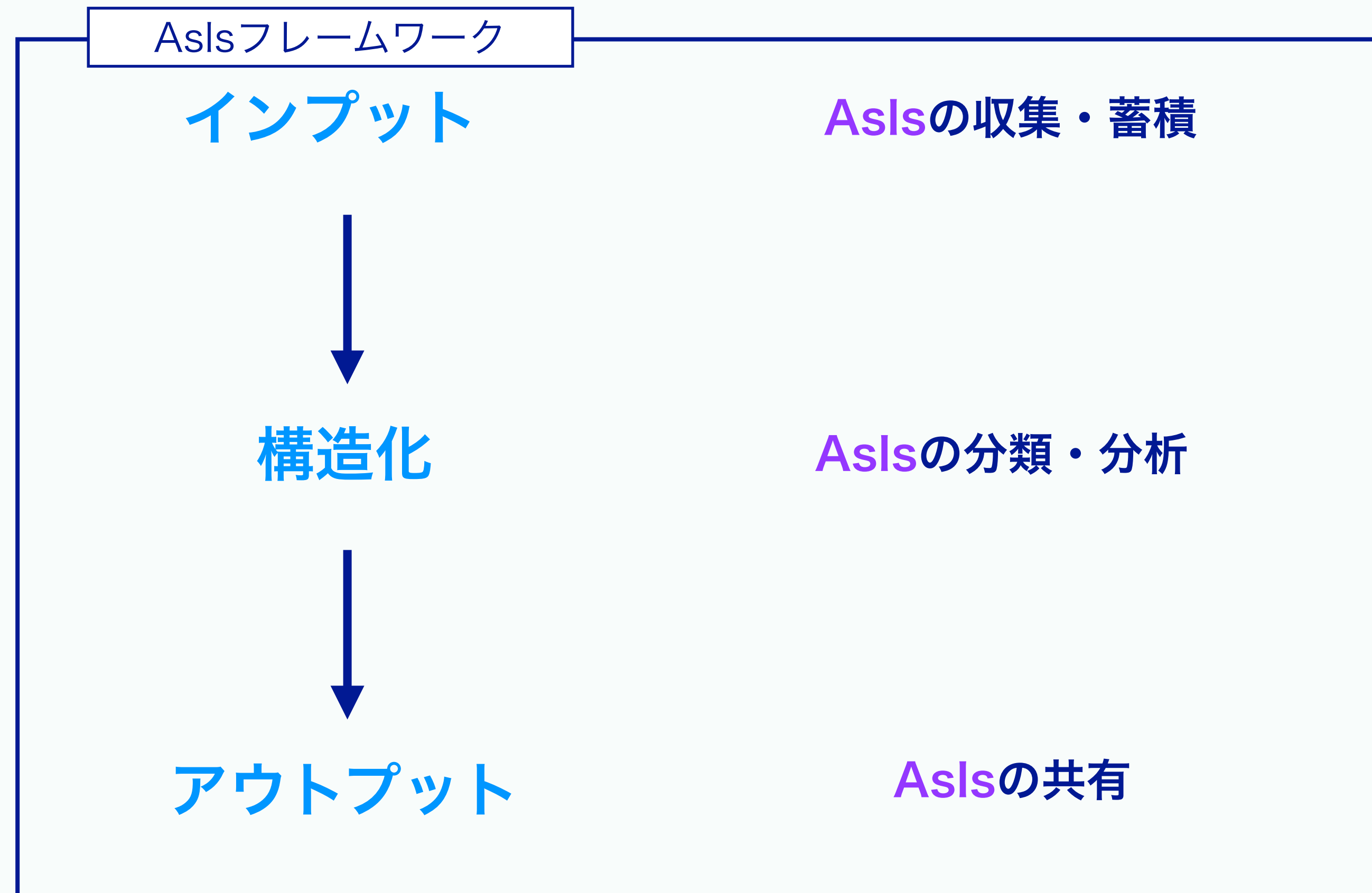
運用現場のAsIsの収集と蓄積

運用現場のAsIsを運用現場と設計側で共有

運用現場のAsIsの継続的な分析

既に運用が動いている場合は、一番最初にやるべきこと。

Aslsを知るためのサブフレームワーク



重要! 課題を見つけても、いきなり対応しようとするしない。

今回の学習ポイント (再)

インプット

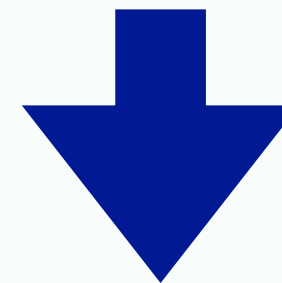
本講義で意識してほしいこと

- ・ 運用現場に多くの課題があり、現場がなかなか楽になっていかないのはなぜなのか。
- ・ 運用現場においてAsIsの認識のギャップが起こりやすいのはなぜか。

一次アウトプット

本講義から持ち帰ってほしい事

- ・ 「運用のAsIs」を把握する上での考え方
- ・ 「運用のAsIs」を分析する上での考え方
- ・ 「運用のAsIs」を共有する上での考え方



- ・ 運用の**AsIs**を把握・分析・共有するための仕組み作り
- ・ 自分達の「運用のAsIs」は**今現在どうなっているか？**

最終アウトプット

今後の予定

